EXATRON MM-800 64K RAM BOARD

CONSTRUCTION, INSTALLATION AND OPERATION

June 1982-

Written by Bob Sawyer, Jim Howell and Jim Perry

Edited by Jim Perry
Graphics by Jim Howell and Alan Gael

exatron
excellence in electronics

This manual is placed in the public domain.
Reproduction in whole or part is expressly authorized.

Exatron Corporation
181 Commercial Street
Sunnyvale, California 94086

(800) 538-8559 (Outside CA)
(408) 737-7111 (Inside CA)

CONTENTS

INTRODUCTION 2

IMPORTANT NOTES 3

INSTALLATION 4
What You Need 4
Fig. 1: Part Location aid 6
Fig. 2: MM-800 Positioning 6
Fig. 3: Modifications To TRS-80 Base 7

OPERATION 8
Table: Command Options 8
Examples 8
Memory Changes 10
Program 1: ROM Into RAM 11
Program 2: Lowercase Driver 11

HOW IT WORKS -- THEORY 12
Memory Expansion 12
Fig. 4: Block Diagram 13
MM-800 Control Section 14
Fig. 5: Control Logic 15
Port Decoding 16
Fig. 6: Port Decoder Circuit 16
Control Status 17
Fig. 7: Status Latch Circuit 17
Control Latch 18
Fig. 8: NMI RESET Circuit 18
Fig. 9: Control Latch Circuit 19
Clock Speed-Up 20
Fig. 10: Clock Speed-Up Circuit 20
RAM / ROM Control 21
Fig. 11: ROM / RAM Switching Circuit 22
Memory-Mapped I/O Control 23
Fig. 12: Memory-Mapped I/O Circuit 23

WARRANTY INFORMATION 24
REPAIR INFORMATION 24
CONSTRUCTION TIPS 25
Circuit Board Modifications (Rev. B) 25
Fig. 13: Rev. B Modification #1 25
Fig. 14: Rev. B Modification #2 25
Fig. 15: Rev. B Modification #3 26
Fig. 16: Rev. B Modification #3 Detail 26
CONSTRUCTION HINTS 27
Fig. 17: Discrete Resistor Networks 27
Fig. 18: Pin Installation 27
Fig. 19: Component Orientation 27
PARTS LIST 28
Fig. 20: Part Placement On PCB 29
Fig. 21: Schematic 30

INTRODUCTION

The MM-800 64K RAM board can truely unleash the full power of your TRS-80, the possibilities are endless.

By putting the operating system in RAM you can reconfigure your computer virtually however you wish. Several programs, designed to take full advantage of a 64K TRS-80 and Stringy Floppy storage, are under development at Exatron. A word-processor, Forth, CPM, etc., are just some of the programs due soon.

If you have any comments about this manual, either positive or negative, please write and tell let me know -- and I'll incorporate the ideas into the next revision.

The MM-800 is an elegant design, and relatively easy to install, but you must be able to solder neatly -- if you have any reservations about your abilities then don't try it, get someone else to install it for you.

Happy 64King!

Jim Perry,
Editor

INTRODUCTION

The MM-800 64K RAM board can truely unleash the full power of your TRS-80, the possibilities are endless.

By putting the operating system in RAM you can reconfigure your computer virtually however you wish. Several programs, designed to take full advantage of a 64K TRS-80 and Stringy Floppy storage, are under development at Exatron. A word-processor, Forth, CPM, etc., are just some of the programs due soon.

If you have any comments about this manual, either positive or negative, please write and tell let me know -- and I'll incorporate the ideas into the next revision.

The MM-800 is an elegant design, and relatively easy to install, but you must be able to solder neatly -- if you have any reservations about your abilities then don't try it, get someone else to install it for you.

Happy 64King!

Jim Perry,
Editor

IMPORTANT NOTES

A: READ ALL OF THIS MANUAL BEFORE DOING ANYTHING ELSE!!

B: YOU WILL VOID ANY RADIO SHACK WARRANTY BY INSTALLING THE MM-800.

C: IF YOU ARE NOT ABSOLUTELY SURE THAT YOU CAN SUCCESSFULLY INSTALL THE MM-800 -- DON'T TRY!!

GET A GOOD ELECTRONICS TECHNICIAN TO DO IT FOR YOU -- OR RETURN UNUSED UNIT FOR A FULL REFUND.

D: BARE BOARD & KIT BUILDERS

THE MOUNTING PINS MUST BE INSTALLED FIRST -- EXATRON CAN INSTALL THEM FOR ABOUT $30, CALL FOR A PRICE.

NEXT INSTALL ALL THE RESISTORS & CAPACITORS, FOLLOWED BY THE IC SOCKETS.

THEN THE IC'S. DOUBLE-CHECK CAREFULLY FOR SOLDER BRIDGES ETC.

DON'T FORGET TO DO THE THREE MODIFICATIONS FROM PAGES 26 AND 27.

INSTALLATION

You you will need:

Soldering Iron and good soldering technique
Solder
Phillips Head Screwdriver
X-Acto knife
Wire

1) Power down your computer and disconnect all peripherals. Remove the cassette, video, and power connections from the back of the computer.

2) Find a clear work space, installation will take about 1 hour.

3) You may wish to put a towel or foam pad under the your computer while you work.

4) Turn your computer over onto its keyboard and remove the six cross-head screws that hold it together.

Remember where the screws come from, as they are of three different lengths.

5) Turn your computer back over, so it is sitting on its feet again. Carefully remove the top half of the molded case.

You will see your keyboard and the bottom of the computer board are being held apart by a combination of molded plastic standoffs and white plastic spacers. Note where the spacers come from, and remove them.

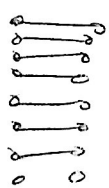
6) You want to take the keybaord and the computer board out from the bottom of the molded case. The computer and keyboard are connected together by a two-inch flexible cable.

The connections on this cable are extremely delicate, so try not to flex it very much.

7) You should now have both the keyboard and component side of the computer board facing you.

Carefully remove Z40, (the Z80 CPU chip), the eight memory chips (Z13-Z20), and the DIP shunt (X3). Be careful in removing all pieces, but be especially careful with the Z80 CPU.

It may help to put a flat-blade screwdriver under one side of the CPU, and pry it up slowly a little bit at a time, from each end, until it is free from the socket.



8) Install the CPU into the socket provided on the MM-800 board -- make absolutely certain that it is the correct way round!!!

Note: It is recommended that a Z80A be used instead of the standard CPU because of its faster operating speeds. If the 64K RAM chips have not been installed yet, install them now.

Now find the jumper select area on the MM-800 near X3. The jumper should be in position B to A if you have a three-chip ROM set. If the your computer has a blank IC socket in Z34, and a cable going from Z33 to a small board taped to the back of the computer, you probably have the three-chip ROM set. If you have a two-chip ROM set then the jumper should be put in position B to C.

9) For basic operation of the MM-800 two wires on the computer board, and two wires from the computer board to the MM-800 are required. These are:

Length	From	To
3"	Z51 (74LS157) pin 13	Z38 (74LS367) pin 9
3"	Z51 (74LS157) pin 14	Z38 (74LS367) pin 11
3.75"	Z51 (74LS157) pin 12	MM-800 connection A
13"	Z56 (74LS92) Pin 11	MM-800 connection H

For the memry-mapped I/O option two extra wires, and a trace cut on the TRS-80, are needed. They are:

1.5"	Z36 (74LS32) pin 4	MM-800 connection B
1.5"	R48 (Near CPU *)	MM-800 connection C

* See page 23 for details

See figure 1, page 6, for general locations of parts referenced here.

It is recommended that the wires on the computer board be soldered before attempting to install the MM-800 board into the open sockets. After the MM-800 has been installed solder the wires from the TRS-80 to the MM-800.

10) Now place the MM-800 board over the sockets so the board may be pushed into the open sockets. Check the alignment of the pins to the sockets and the straightness of the pins to each other and the board. You do want any missplaced or bent pins when trying to push the board into the sockets! See figure 2, page 6, for an idea of how the MM-800 fits in place.

Remember that taking the board out once it has been pushed into the sockets is not very easy!

When pushing the board into the sockets place your thumbs on the top of the board and your remaining fingers under the computer board under the MM-800. This will help reduce the stress on the MM-800 and computer also.

After seating the MM-800 into the open sockets check to make sure all the pins are seated correctly, and are all pushed completely into the sockets.

11) You now need to modify the pins on your molded bottom half of the case for the computer. Figure 3, page 7, shows a complete bottom case with the three pins that need to be modified marked A, B and C. To shave the sides of posts B and C an X-Acto knife is perfect.

Be careful and don't cut yourself.

12) Follow steps 4 through 7 in reverse order to put your computer back into its case.

13) Installation of the MM-800 has now been completed and now you can test it by entering program 1 from page 11.

Figure 1: Locations of parts mentioned in installation procedure.

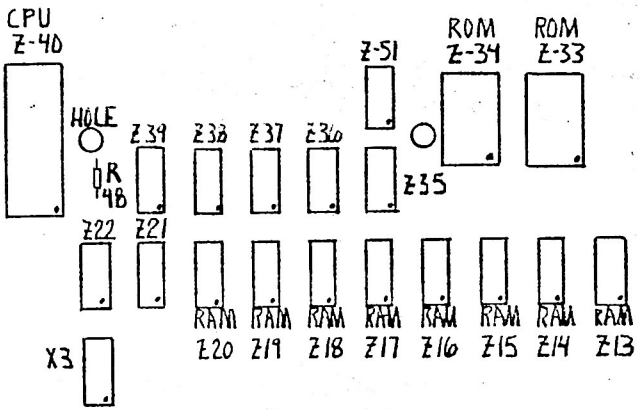


Figure 2: How the MM-800 board fits in place.

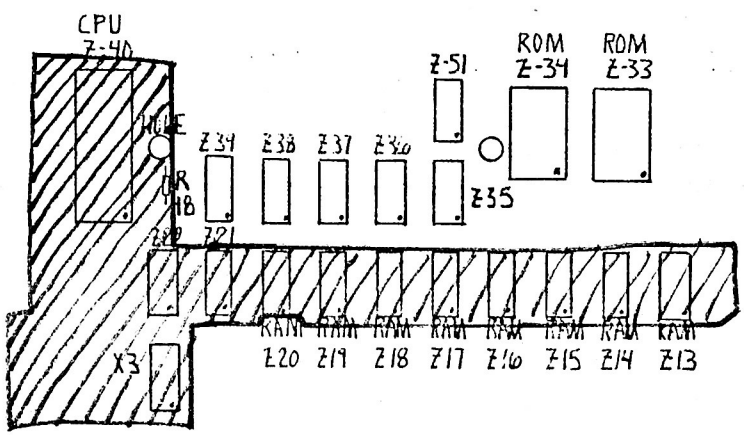
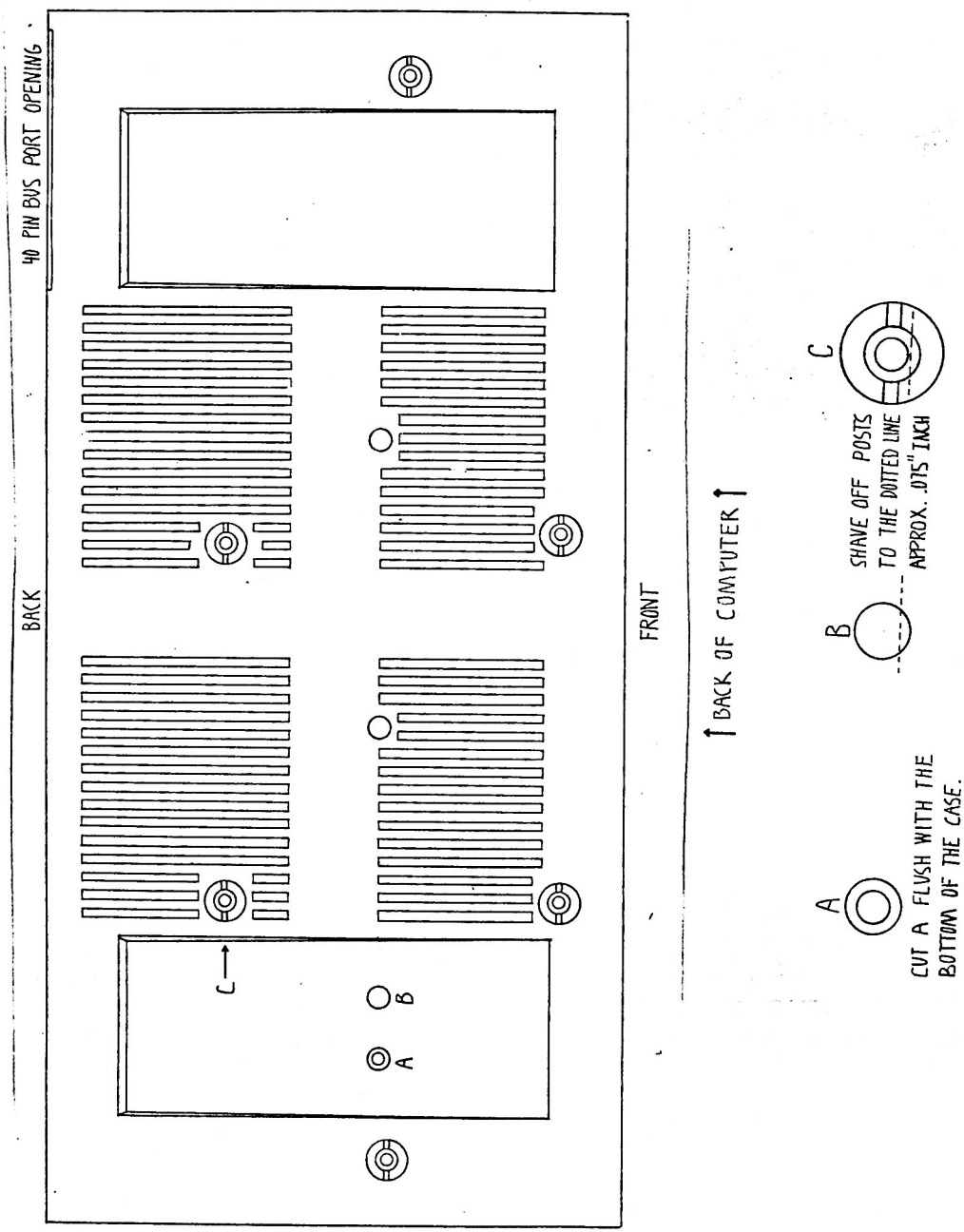


Figure 3: Modifications to the TRS-80 base.



OPERATION

Operation of the MM-800 is really quite straight-forward. When power is first applied to the TRS-80, the MM-800 is already working. The first thing you should notice is that it takes longer to initialize. This is because you now have 48K of memory online. From BASIC enter

? MEM <Enter>

and see those LARGE numbers come up.

The following table lists the command options of the MM-800.

DATA BIT	FUNCTION	WEIGHT	SET	RESET
0	SPEED-UP	1 (01)	FAST	NORM
1	NOT USED	2 (02)	-	-
2	NOT USED	4 (04)	-	-
3	NOT USED	8 (08)	-	-
4	RAM/ROM	16 (10)	RAM	ROM
5	RAM/MM I/O	32 (20)	RAM	MM I/O
6	SPARE	64 (40)	ON	OFF
7	NOT USED	128 (80)	-	-

If an OUT instruction, specifying the above bits, is sent to Port 254 then the corresponding action will take place.

REMEMBER - ALL THE BITS ARE CHANGED AT THE SAME TIME.

EXAMPLES

Lets assume that you want turn the speed-up option ON, the following BASIC instruction could be executed:

```
10   OUT 254,1
```

Then at some later time if you wish to turn on the Spare Latch, to control some device of your own, you could issue the BASIC command

```
80   OUT 254,64
```

This would certainly turn on the Spare, but it would also turn off the speed-up. Here's the reason why ...

```
      Data bit 7 6 5 4 3 2 1 0
OUT 254,1   = 0 0 0 0 0 0 0 1   Speed-up ON all others off.
OUT 254,64  = 0 1 0 0 0 0 0 0   Spare ON all others off.
```

To keep the speed-up ON the value should have been 65; the value for the speed-up plus the value for the Spare (1 + 64).

To give you the ability to dynamically change states in your system, we have included a feature in the Control Port that allows you to read it's status from software. To read the Port all that is necessary is to issue an INPUT instruction to Port 254. In BASIC this would appear as:

```
30 V = INP(254)
```

After this command is issued "V" will have the value contained in Port 254. If all bits are OFF the value would be 142, this is because the unused bits are read as being ON. For example:

```
Data bits 7 6 5 4 3 2 1 0
          1 0 0 0 1 1 1 0 = 128 + 0 + 0 + 0 + 8 + 4 + 2 = 142
```

You could set up a table in an array in order to decode the appropriate bits, but this would be both time consuming and rather awkward. It would be much easier (at least in my opinion) to use the logic functions "AND" and "OR" to reset and set the bits respectively. Using the above example the following sequence could be used:

```
10 GOSUB 2000            'GET THE PORT STATUS
20 V=V OR 1              'SET THE SPEED-UP BIT
30 OUT 254,V             'PUT IT IN THE CONTROL LATCH

CONTINUE PROCESSING .......

70 GOSUB 2000            'GET THE PORT STATUS
80 V=V OR 64             'SET THE SPEED-UP BIT
                         'LEAVING THE OTHERS ALONE !
100 OUT 254,V            'PUT IT IN THE CONTROL LATCH

CONTINUE PROCESSING .......

200 GOSUB 2000           'GET THE PORT STATUS
210 V=V AND NOT 32       'RESET THE SPARE BIT
                         'LEAVING THE OTHERS ALONE !
220 OUT 254,V            'PUT IT IN THE CONTROL LATCH

CONTINUE PROCESSING .......

2000 V=INP(254)          'GET THE PORT STATUS
2010 V=V AND 113         'MASK OUT THE UNUSED BITS
                         ' BITS 0 1 2 3 4 5 6 7
                         '      0 1 1 1 0 0 0 1
                         ' SAVE THE ONES THAT COUNT !
2020 RETURN
```

Using this method you don't need to worry about keeping track of the bits that are set or reset, only about what the program needs at the time.

MEMORY CHANGES

The two bits that control RAM require that the programmer has some basic skills. Although it is possible to change the ROM and RAM control latches from BASIC, unless you have put something in that area there can be only the heartbreak of a locked up system. Once you change bit D4, and BASIC tries to find the next command, all that is there is random garbage. Usually this results in "@9" 's all over the screen.

Example program 1, on page 11, demonstrates a method for reading the ROM into high RAM, then makes the switch from ROM to RAM, and reloads the ROM data into the new low RAM. At this point control is turned over to BASIC and the user continues as before.

The prompt character is also changed to a colon ":", and the READY message is erased. You can use this technique to create your own "custom" BASIC. You can use a similar method for putting your own operating system in the area once occupied by ROM.

There are some precautions to be taken when changing out the memory-mapped I/O area for RAM. This is the area used by the keyboard and video, so when its being used as RAM the above areas are effectively turned off. It will take some skilled programming to use this area effectively. It does offer some interesting posibilities though -- how about using this area for keeping extra video screens?

Both example programs can be entered using the Editor/Assembler for the machine language version, or directly in BASIC for the BASIC versions.

Program 1: Loads ROM into RAM and passes control to RAM.

```
 5 GOTO 200
10 ' THIS PROGRAM WILL MOVE THE DATA CURRENTLY IN ROM
20 ' TO RAM, AND THEN WILL RETURN TO THE PROGRAM.
30 ' FOR MACHINE LANGUAGE BUFFS HERE IS THE CODE..
40  010030      LD      BC,3000H        ;COUNT
50  110080      LD      DE,8000H        ;TO AREA
60  210000      LD      HL,0            ;FROM AREA
70  EDB0        LDIR                    ;MOVE DATA
80  3E10        LD      A,10H           ;RAM/ROM
90  D3FE        OUT     (0FEH),A        ;SWITCH
100 EB          EX      DE,HL           ;SWAP TO/FROM
110 010030      LD      BC,3000H        ;COUNT
120 EDB8        LDDR                    ;MOVE AGAIN
130 3E3A        LD      A,3AH           ;NEW PROMPT
140 32771A      LD      (1A77H),A       ;STORE IT
150 3E00        LD      A,0             ;CLEAR A
160 C9          RET                     ;RETURN
170 ' IF THIS PROGRAM IS ENTERED WITH EDTASM
180 ' CHANGE THE LAST LINE TO :
190 C3191A      JP      1A19H
200 DATA 01,00,48,17,00,128,33,00,00
210 DATA 237,176,62,16,211,254,235,01,00,48
220 DATA 237,184,62,58,50,119,26,62,00,201
230 FOR I = 28672 TO 28700
240 READ A
250 POKE I,A
260 NEXT I
270 POKE 16526,00
280 POKE 16527,112
290 X=USR(0)
300 PRINT "NOW OPERATING IN RAM"
```

Program 2: Lowercase driver

```
10 CLS: GOTO 1000
20 ' SO YOU WANT LOWER CASE...
30 ' WELL HERE IS A LITTLE ROUTINE THAT WILL NOT ONLY ENABLE
40 ' LOWER CASE BUT WILL ALSO MAKE THE SCREEN MEMORY CONTAIN
50 ' TRUE ASCII VALUES.
60 ' NOTE:
70 ' SOME PROGRAMS THAT INTERCEPT THE KEYBOARD/DISPLAY WILL NOT
80 ' WORK BECAUSE WE HAVE INTERCEPTED THESE FUNCTIONS IN THE ROM
90 ' AREA AND THAT WAS NOT ANTICIPATED BECAUSE IT COULDN'T BE DONE
100 ' UNTIL NOW THAT IS ....
1000 POKE 1142,00 'Old value is 64 (40H)
1010 POKE 1148,00 'Old value is 32 (20H)
1020 PRINT "Do you want the SHIFT key normalized ? <Y/N>"
1030 PRINT "(normal lower case, shifted upper case)"
1040 IN$=INKEY$
1050 IF IN$ = "" GOTO 1040
1060 IF IN$ = "y" OR IN$ = "Y" THEN POKE 1048,56: END
1070 IF IN$ = "n" OR IN$ = "N" THEN POKE 1048,48: END
```

THEORY OF OPERATION

For this section of the manual you will need to refer to the accompanying schematics and simplified drawings. It will also be helpful if you can refer to the Radio Shack technical reference handbook to see the interconnection points within the TRS-80.

If you only care that the system works, and are not interested how, then skip this section -- turn on the power and enjoy the many benefits of your MM-800.

MEMORY EXPANSION

Lets begin by looking at the block diagram (figure 4, page 13). The standard 48K memory expansion is shown on the far right of this drawing. You will notice the boxes labeled "MM-800 48K RAM CONTROL" and "MM-800 48K RAM". These boxes are the part of RAM that work as the "standard" 48K system memory. This memory is always enabled and operates as if you had an expansion interface connected.

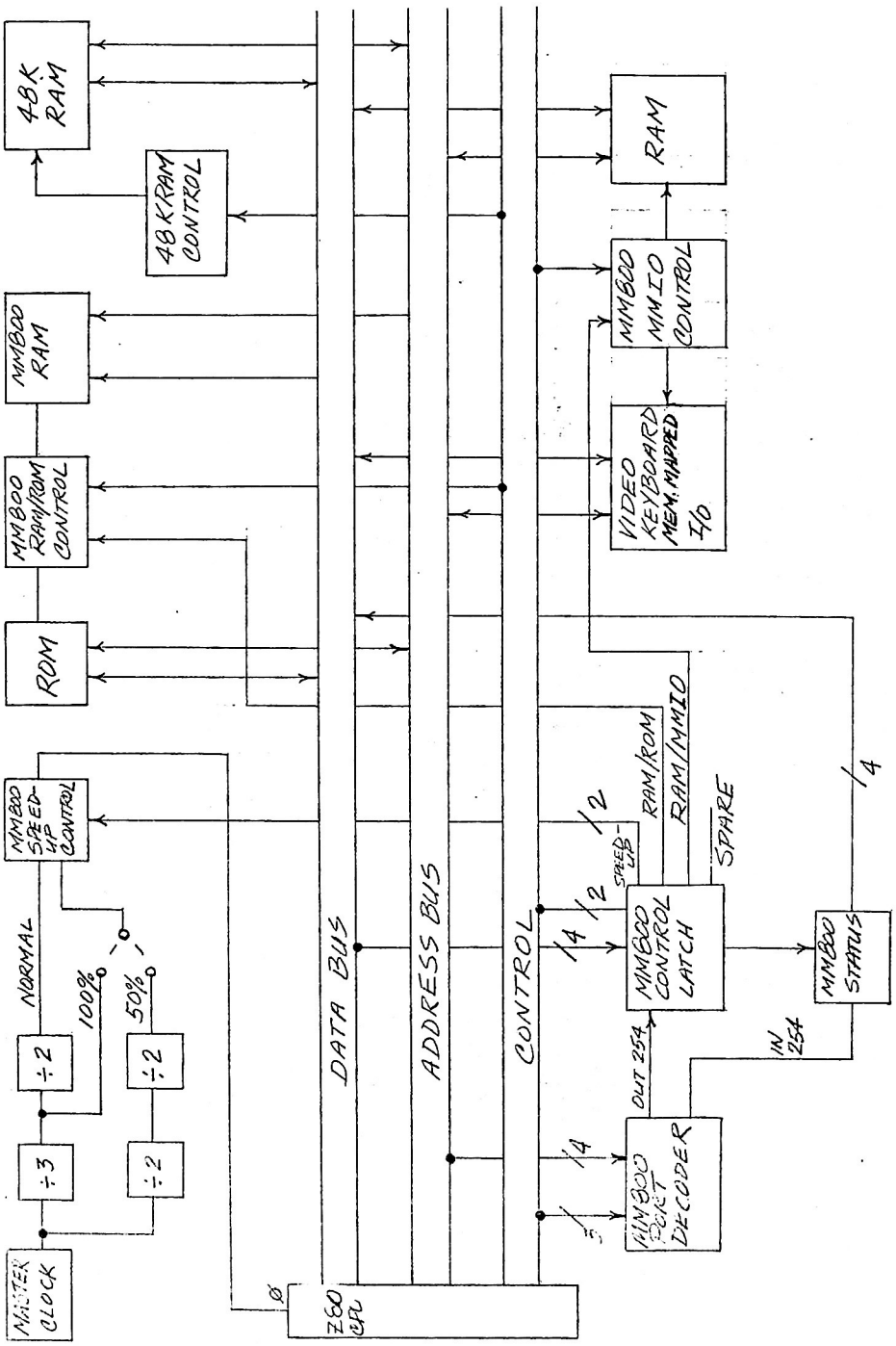
The term "48K RAM" is slightly misleading. We use this term to signify the part of memory that simulates the expansion interface. It's actually, as is all RAM in the MM-800, part of a 64K block of memory made up from eight 64K memory chips. These can be seen on the bottom of the MM-800 schematic at the back of this manual.

The RAM chips are dynamic NMOS components organized as 65,536 one-bit data cells. Because of this eight of them required, one chip for each of the 8 data bits. These have been conveniently labeled as D0-D7 to match the data bits that each will hold. In the TRS-80, as in most microcomputers, D0 is the least significant bit (LSB). Pin 2 of the RAM chip is for data input, and pin 14 is for data output.

Addressing the 64K RAMs is done in the same way as the 16K chips, that you removed from the keyboard, were addressed with one difference, 16 address bits are required insted of 14. These bits are shown on the left side of the schematic and are labeled MA0-MA7. This stands for "Multiplexed Address" bit 0-7. The TRS-80 supplies MA0-MA6 and MA7 is generated by the jumpers that were installed on the TRS-80 board from Z-38 to Z-51. These two jumpers allowed the TRS-80's address multiplexer to turn address bits 14 and 15 into MA7. The other jumper fcom Z-51 brings the MA7 bit up to the MM-800 board.

On the right side of the schematic are the last remaining signals required for the "standard" 48K RAM. The first of these is WR*. If this signal is a logic (0) then you are going to WRITE into the RAM, and if it is a logic (1) then you will READ from it.

Figure 4: MM-800 Block Diagram



RAS* and CAS* perform several functions. First, they are used to select the address for the cell we wish to access. As you will recall we said that 16 address bits were required to access 64K of storage. You also recall that we only mentioned MA0-MA7. The eight ROW addresses are applied to MA0-MA7 first, and are enabled when the RAS* signal is at a logic (0) level. Next the eight COLUMN address bits are applied to MA0-MA7, and are enabled when the CAS* signal goes to a logic (0).

This operation selects the proper address for the eight RAM chips. It should be noted that BOTH of these signals need to be present to allow RAM to be accessed. This principal will be used later for selecting or disabling other parts of RAM.

Another function of RAS* is to select the appropriate part of RAM to be refreshed. Dynamic RAMs "forget" things if they are left to their own devices, and so they must be "refreshed" occasionally to prevent this. The Z80 CPU provides the address for doing this and the RAS* signal (when used without CAS*) will cause this refresh address to be applied to the RAM. It should be stated here that the Z80 only supplies 128 refresh adresses, and only those devices that use "128 refresh cycles" should be used (like the 64K RAMs supplied by Exatron).

MM-800 CONTROL SECTION

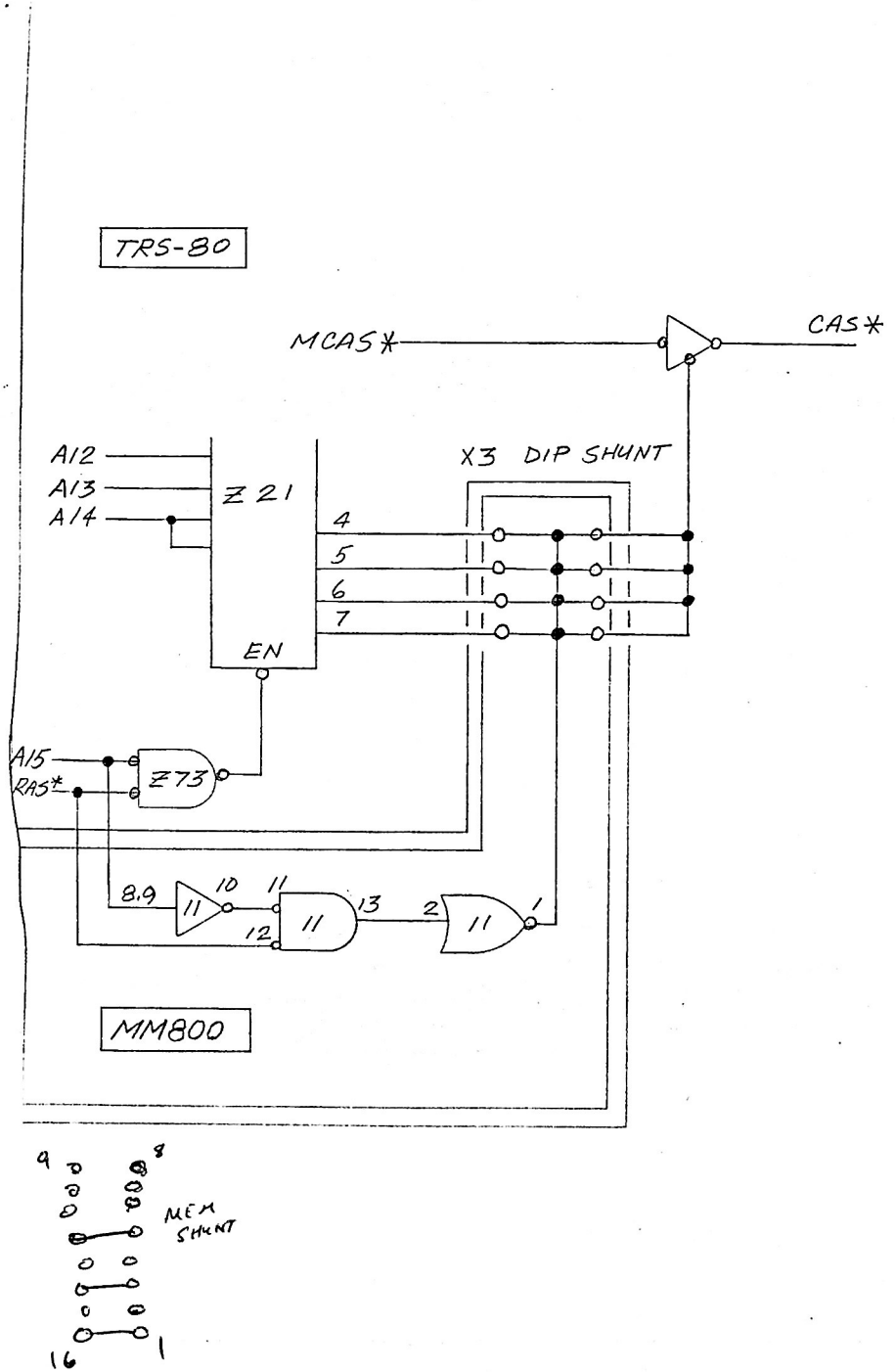
Once again look at the block diagram, immediately to the left of the MM-800 RAM box is the MM-800 48K control logic. It will be easier to describe this section by looking at figure 5, page 15.

The TRS-80 actually does most of the work for this circuit. When A15 is a logic 0, the memory is in the lower 32K. This bit is ANDed with with RAS* to enable Z21, the decoder that selects the TRS-80 ROM, and the memory-mapped I/O area, and the 16K RAM. This decoder divides the lower 32K into eight 4K slots as follows :

Slot 0,1,2 are the ROM (assuming Level II)
Slot 3 is the video, keyboard, and memory-mapped I/O
Slot 4,5,6,7 are the 16K RAM

This division is accomplished using address bits A12-A15. As you can see in the diagram the outputs for 4, 5, 6, and 7 are tied together, enabling the CAS* signal to be sent to the memory chips (remember we said that CAS* had many uses, one of them is to enable memory). The remaining part of figure 5 shows the MM-800's part in all of this. To enable the upper 32K of storage in our 64K RAM chips all that is needed is to "AND" address bit A15 with RAS* and make it also enable the CAS* signal. This is done by connecting into the DIP shunt on the TRS-80 as shown.

Figure 5: MM-800 Control Logic

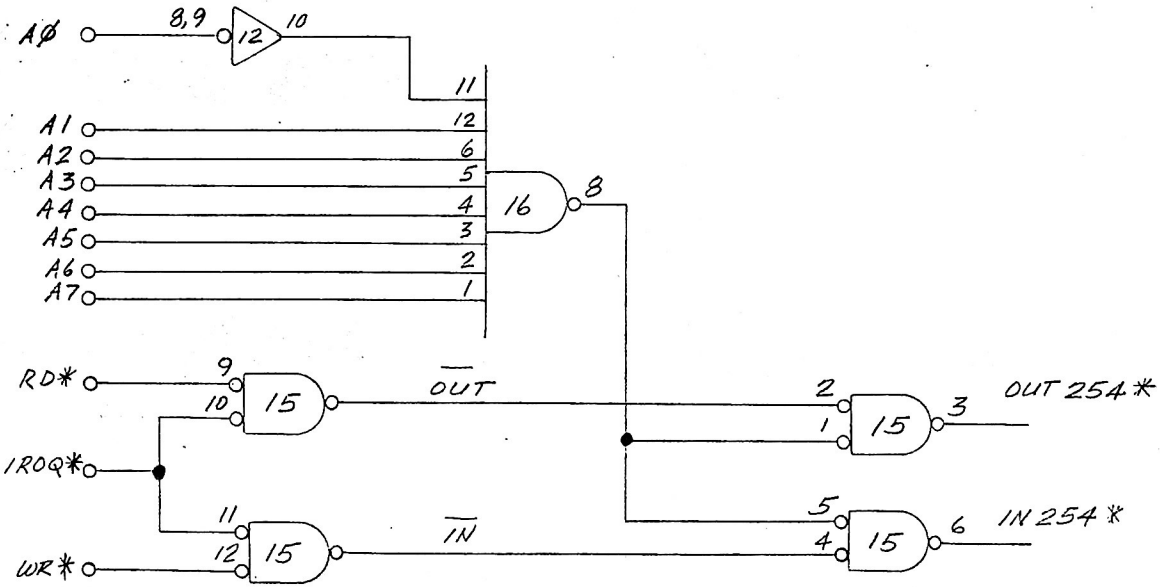


PORT DECODING

The 48K memory section of the MM-800 is an "automatic" function, by that we mean that it doesn't require any concious effort on your part to use it -- it is just there, sort of like taxes. The remaining sections do require operator intervention however, since you may or may-not want them to be active at any given moment. This means that there must be a way for you to communicate your wishes to the hardware. The section at the lower left of the block diagram is where this happens.

The MM-800 Port Decoder is used to decode the Port address of 254 (FE Hex) and then select whether the signal is IN or OUT. The actual circuitry can be seen in figure 6.

Figure 6: MM-800 Port Decoder Circuitry



The address is decoded by 8-input AND gate (IC16), and an inverter (IC12), the output is a logic 0 when the address bits A7-A0 equal 254.

When the Z80 CPU sends IORQ* and RD* at a logic 0 then an IN command has been issued. An IORQ* and a WR* at a logic 0 are the result of an OUT command.

These signals are decoded by the AND gates on the left and the resultant signal is ANDed with the 254 address decode to give us IN254* or OUT254* respectively.

In the block diagram we see these two signals being sent to the MM-800 Control Latch and Status boxes. Lets take the Status box first.

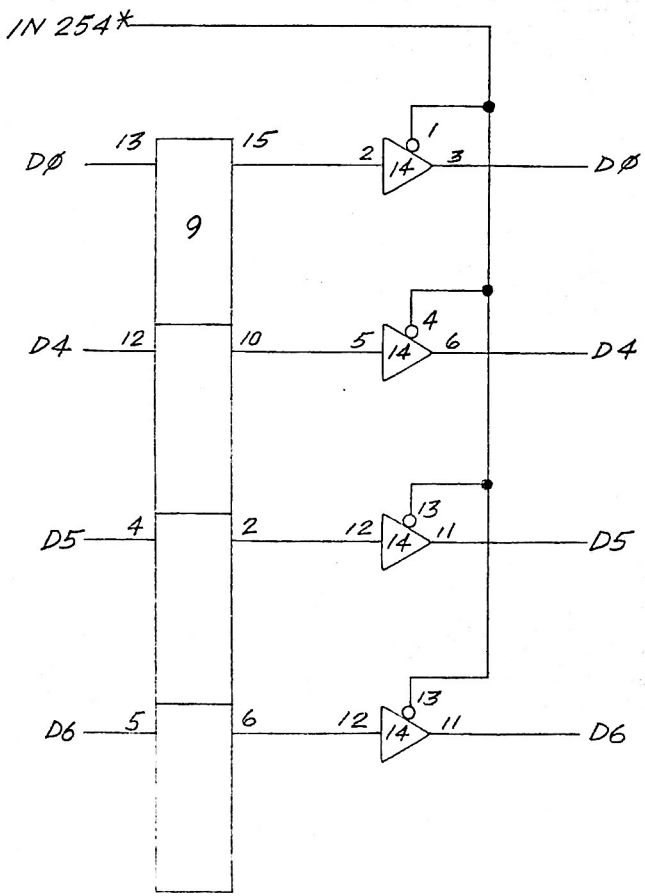
CONTROL STATUS

Figure 7, below, shows that when the IN254* signal is at a logic 0 the four gated buffers are enabled and allow the values present at the control latches to be placed on the data bus, as shown below:

Data Bit	Function	(1)	(0)	Value
0	SPEED-UP	ON	OFF	1
4	RAM/ROM	RAM	ROM	16
5	MM I/O	RAM	MM I/O	32
6	SPARE	ON	OFF	64

This is one of the unique and extremely useful functions of the MM-800, it allows your software to read the latch settings, and determine what action to take based on the value read.

Figure 7: Status Latch Circuitry.

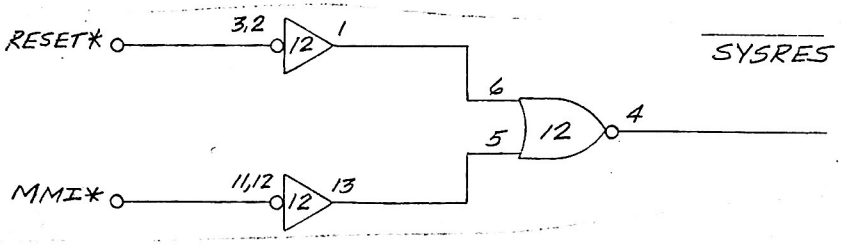


CONTROL LATCH

The Control latch does exactly as its name implies, it controls the operator selectable parts of the MM-800. Its control covers whether you operate in RAM or ROM, whether you are using the keyboard / display / memory-mapped I/O area / or RAM, the CPU clock speed, and the spare latch. Because it has such a great amount of control over your system, someone has to control it ... and that someone is you.

When you first apply power, the TRS-80 sends a RESET* signal to system, this signal is intercepted by the MM-800 and resets all the latches, assuring that the system comes up in the normal TRS-80 mode. The circuitry for this is shown in figure 8 below. The other signal shown here is NMI*, This is the Non-Maskable Interrupt or RESET button on the TRS-80. This assures that if you reset the computer it will return to the normal TRS-80 mode. An important sidenote here is that if you were operating in RAM (as opposed to ROM) and pressed the RESET button, you will go back to ROM -- the system you had in RAM will still be there however, and if it was operable it can be used by switching back to RAM and returning to the appropriate entry point.

Figure 8: NMI RESET Circuitry.



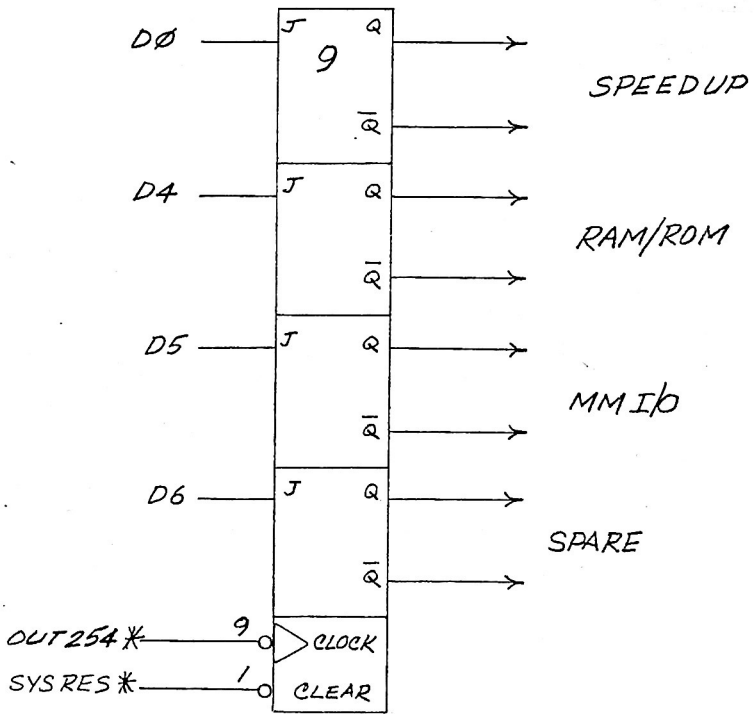
The Control Latch is shown in figure 9, page 19, is a rather simple-looking device, when seen on its own, it is four type-D latches in a single chip (IC9). When the OUT254* signal is sent from the Port Decoder, the values that are present on the data lines will be loaded into the latches. This will either turn on or off the function for that bit. It should be pointed out that all four latches are changed at the same time. For example, if you issue an OUT 254,1 to turn the speed-up ON, and then issue an OUT 254,64 to change to the alternate character set of the Exatron Character Generator, (one of the devices you can control with the Spare latch). What do you think will happen? If you said the CG will be switched and the speed-up will be turned OFF -- you are correct. The value 64 in binary looks like:

0 1 0 0 0 0 0 0

The Spare bit is a logic 1 and the speed-up bit is a logic 0. If you want both the bits on you must send an OUT 254,65.

The best method for controlling these functions is to read the status of the latches first, and then use the logical "AND" and "OR" functions to reset or set the the bits respectively. This is the reason for the Status latch being there in the first place !

Figure 9: Control Latch Circuit.



CLOCK SPEED-UP

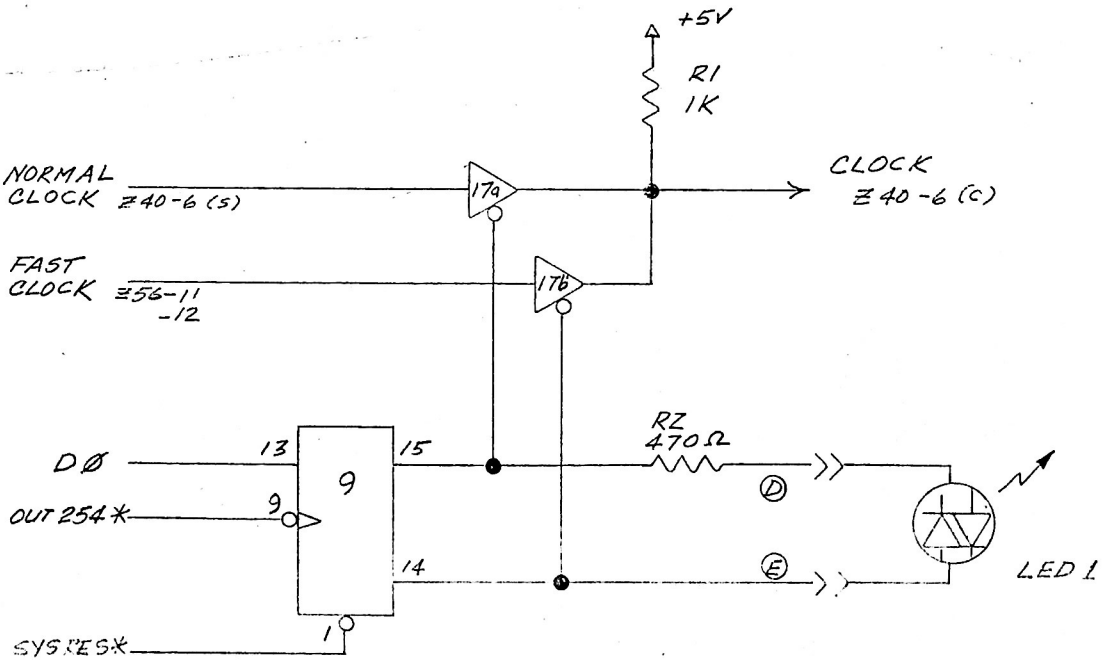
The TRS-80 master clock and clock dividers are shown in the upper left on the block diagram. Normally the master clock is divided by 6 to get the CPU operating clock down to 1.7 MHz. It is a simple change to double that speed by tapping the divide-by-3 circuit. Some TRS-80's will not operate at this speed and therefore the added divide-by-4 circuit will give only a 50% speed increase. We've never seen one that would not operate at this speed.

You may have no trouble at the 100% clock rate and we suggest that you try this way first. If it doesn't work at this speed, you can try installing a Z80A (a high-speed version of the Z80 chip), otherwise you will have to settle for the 50% option.

The MM-800 Clock Control is shown in figure 10. When the OUT254* signal clocks the data into the control latch, if bit D0 is a logic 1 then the fast clock will be turned ON via buffer 17b, and buffer 17a will turn OFF the slow clock. If D0 is a logic 0 the opposite occurs.

LED1 is a tri-color LED available from Radio Shack, and is an optional indication of the clock state. It will be either red or green depending on the clock speed and your taste when installing the LED. R2 is a current limiting resistor on the MM-800 board.

Figure 10: Clock Speed-Up Circuit.



RAM/ROM CONTROL

Once again on the block diagram, if you will look to the right of the Speed Control box, you will see the ROM, the MM-800 RAM, and the MM-800 RAM Control. The ROM is Radio Shack's, and we have already discussed the RAM (it's part of the 64K).

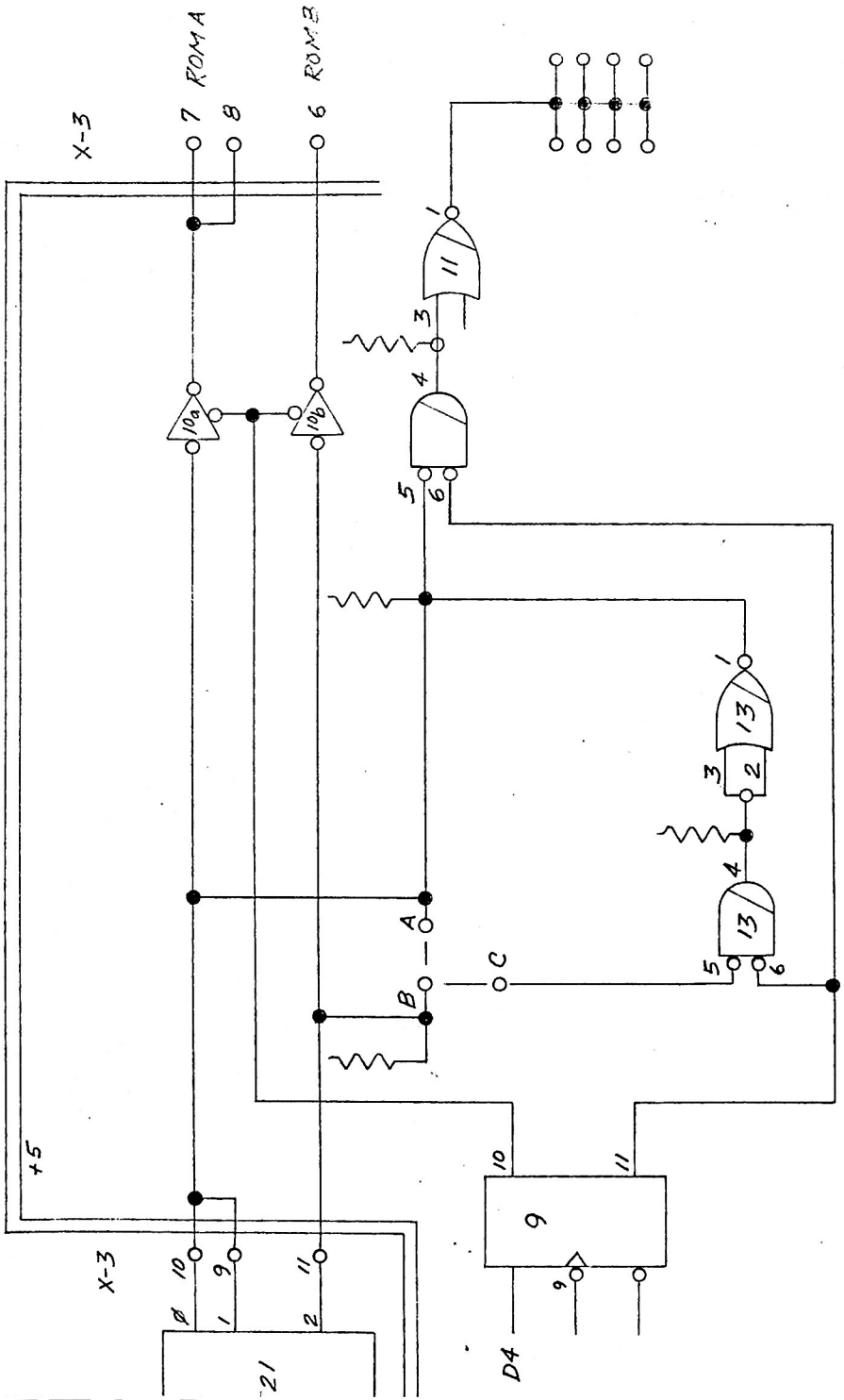
The MM-800 ROM/RAM Control is shown in figure 11, page 22. The major signals to the ROM/RAM control come from Z21, the ROM address control that we discussed before. This is where the TRS-80 decodes which ROM or MM I/O device to communicate with. When you are using the ROM these signals are sent from Z21 to the DIP shunt and then to the appropriate ROM socket.

If you have the two-chip ROM set the signals for the first 2K (0000 to 1FFF hex) go to ROM A, and the signal from the 3rd K (2000 to 2FFF hex) go to ROM B.

If you have a three-chip ROM set then these signals are connected together, by jumpers A and B, which sends the appropriate signals to the ROM connector (actually both connectors). The two buffers 10a and 10b are enabled by the RAM/ROM control latch and D4 (as discussed in the Control Latch section).

If the Control Latch is set to the logic 1 condition, then the ROM buffers will be turned OFF, and the RAM enable gates will be turned ON. If you have the three-chip ROM set then you have jumpered points A and B together, and any access directed to ROM will be intercepted and converted into the CAS* signal. Then it is routed through the DIP shunt, thus sending the READ or WRITE request to RAM. In this case IC13 is effectively turned OFF. IC13 is used when you have the two-chip ROM set. The above procedure holds true here also, a ROM request is sent to the memory chips as CAS*. IC13 was needed to keep the two ROM sockets isolated from each other, yet allowing both signals to activate RAM.

Figure 11: ROM/RAM Switching Circuitry.



MEMORY MAPPED I/O CONTROL

The section of memory between 3000 and 3FFF hex is (or rather, was) reserved for memory-mapped I/O. This is the keyboard, video, disk, printer, Stringy Floppy, and other gizmos. There was no memory here at all. The control for this section (you guessed it -- you can use RAM here also), is shown in figure 12, below.

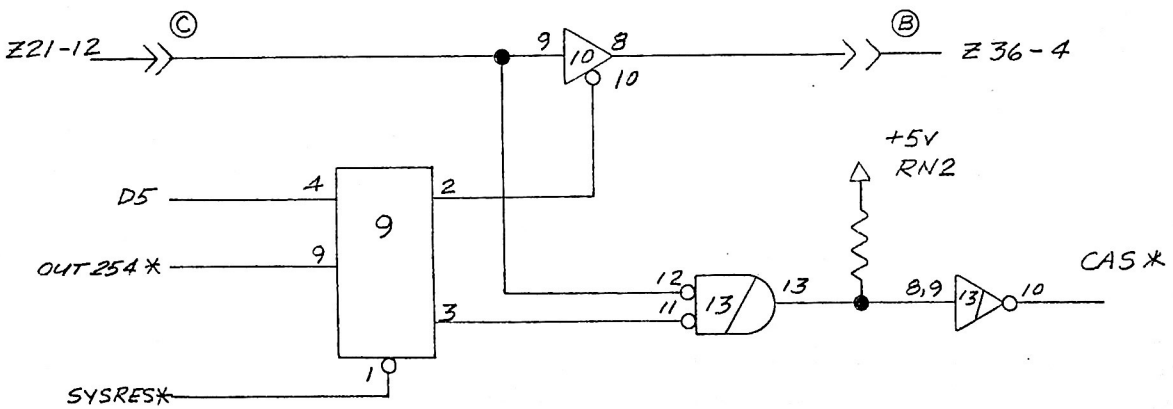
This area is controlled by the Control latch just as the others were, only this time using D5 as the control bit. In order to make use of this section you have to make a trace cut on the TRS-80 circuit board. The trace to cut is between IC21 pin 12 and IC36 pin 4 (under R 48). This is because the signals for this area do not go from Z21 through the DIP shunt as the others do.

SEE ADDITIONS TO MM-80 PROVISIONAL MANUAL

After the trace cut, jumpers are placed from point C on the MM-800 board to Z21-12, and from point B on the MM-800 to Z36-4, this put IC10 in series with the control line that was cut. Now when the Control latch is reset the TRS-80 acts as it always has, but when the latch is set the memory-mapped I/O section is turned off and RAM is being acessed. This is accomplished by IC13 sending CAS* to the RAMs.

This area can be controlled by software, but it must be remembered that when you use this feature, the keyboard, video, etc., are turned off and cannot be used.

Figure 12: Memory-Mapped I/O Control Circuitry.



WARRANTY INFORMATION

Because of the nature of the MM-800 Exatron only makes the following guarantees:

A: BARE BOARDS -- The layout of the MM-800 PCB is free of all defects, apart from the Revision B modifications noted on pages 25 and 26.

B: 64K RAM CHIPS -- The 64K RAMs will be replaced, if handled with normal precautions, if they fail within 1-year of installation.

C: EXATRON ASSEMBLED MM-800's -- These are tested before shipping at Exatron, after successful installation they are guaranteed for 1-year from shipment.

D: Under no circumstances will Exatron be liable for consequential or other damages, caused by the failure or incorrect installation, of any MM-800 board.

REPAIR INFORMATION

If your MM-800 doesn't work call Exatron on the Hot Line.

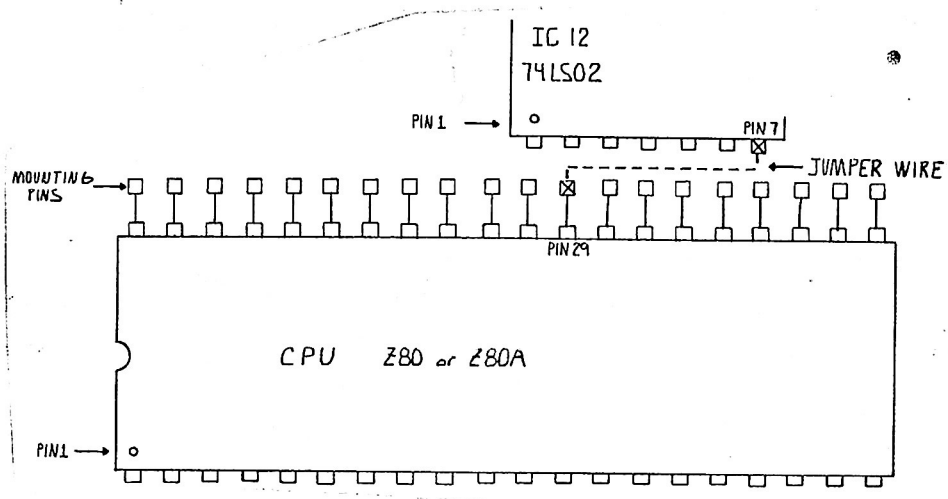
Before returning any unit to Exatron you MUST get a RETURN AUTHORIZATION NUMBER from the Service Department.

CONSTRUCTION TIPS

A. CIRCUIT BOARD MODIFICATIONS (REV B ONLY)

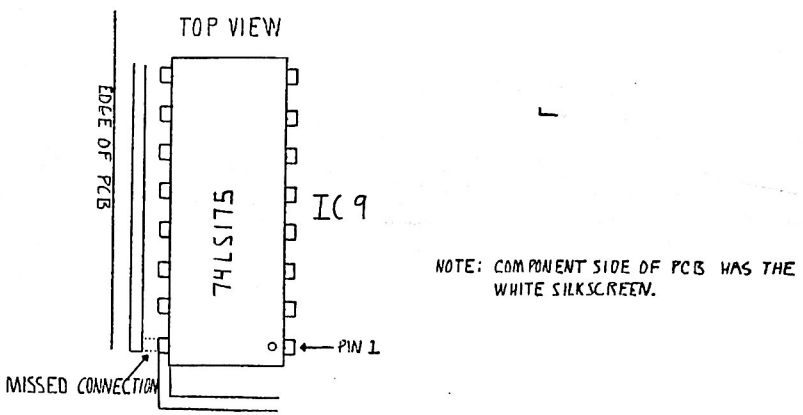
1: Wire a Jumper wire from the Z80 pin 29 to IC12 pin 7 -- as shown in figure 13.

Figure 13: Revision B Modification # 1.



2: Make a solder connection from IC9 pin 16 to trace -- shown in figure 14.

Figure 14: Revision B Modification # 2.



3: Wire a two flexible jumper wires from A to A and B to B, as shown figure 15 & 16.

Figure 15: Revision B Modification # 3.

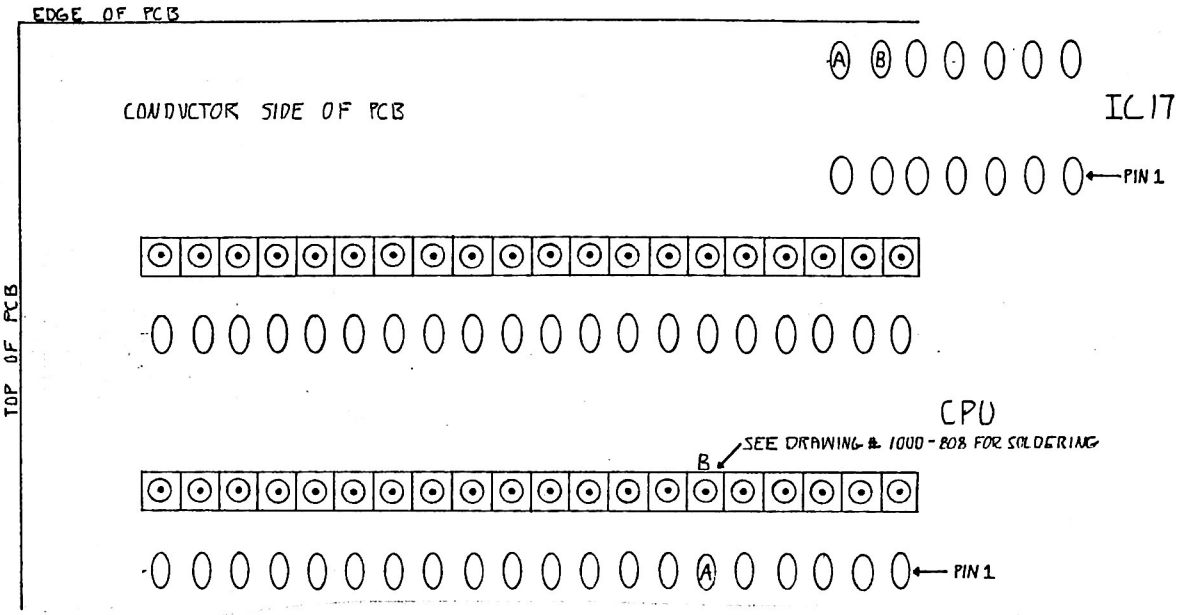
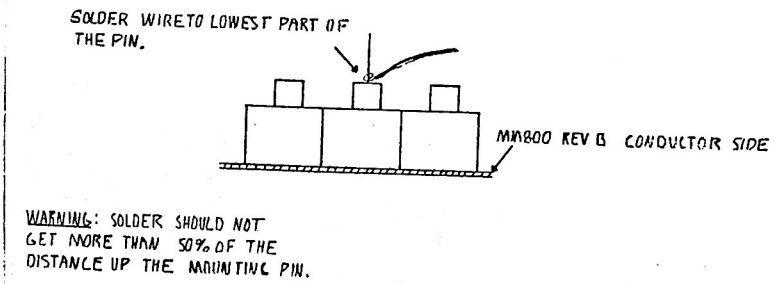


Figure 16: Revision B Modification # 3, detail.


SOLDER WIRE TO LOWEST PART OF THE PIN.
MM800 REV B CONDUCTOR SIDE
WARNING: SOLDER SHOULD NOT GET MORE THAN 50% OF THE DISTANCE UP THE MOUNTING PIN.

CONSTRUCTION HINTS

Figure 17: Making Discrete Resistor Networks.

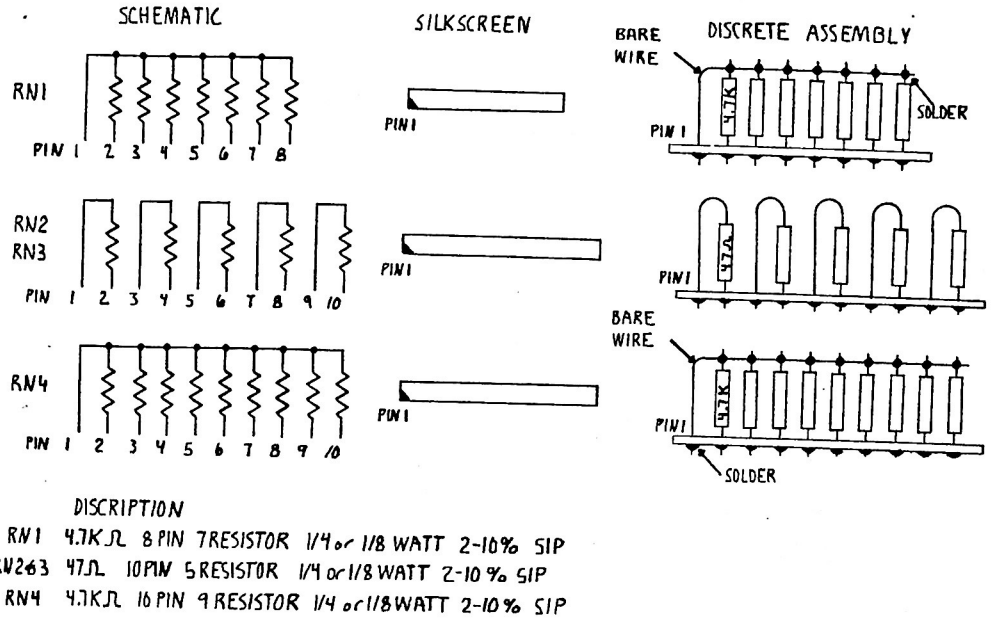


Figure 18: Pin Installation.

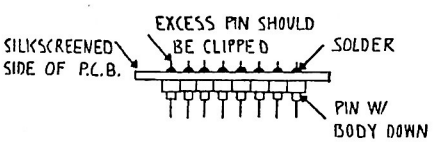
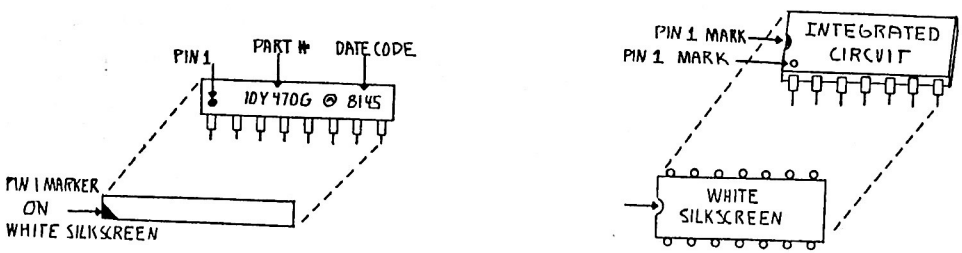


Figure 19: Component Orientation.



PARTS LIST

CAPACITORS

2	C1,2	0.1uF 25V Mono 0.1 lead spacing (Dilectron #2020-Z50-50-104K)
7	C3-9	0.1uF 25V Mono Axial (Plessey #150-100-104K)

RESISTORS

1	R1	330 ohm 1/4W 5% Carbon Film
1	R2	470 ohm 1/4W 5% Carbon Film
1	R3	47 ohm 1/4W 5% Carbon Film
7	R4-10	4.7K ohm 1/4W 5% Carbon Film
1	RN1	4.7K ohm 8-pin Network (Beck 784-1)
2	RN2,3	47 ohm 10-pin Network (Beck 785-3)
1	RN4	4.7K ohm 10-pin Network (Beck 785-1)

SEMICONDUCTORS

8	IC1-8	4864 64K Dynamic RAM 4164 @ $10.ea
1	IC9	74LS175
3	IC10,14,17	74LS125
2	IC11,13	74LS33
1	IC12	74LS02
1	IC15	74LS32
1	IC16	74LS30
1	LED1	TRI-COLOR LED

MISCELLANEOUS

1	PCB	MM-800 PCB
6	6	Mounting-Pin Strips (Samtec TS-120-6AA)
8	8	16-pin low profile IC sockets
1	1	40-pin low profile IC socket